INDIAN INSTITUTE OF SPACE SCIENCE AND TECHNOLOGY
THIRUVANANTHAPURAM

# Assignment #2

Due on 10-09-2014

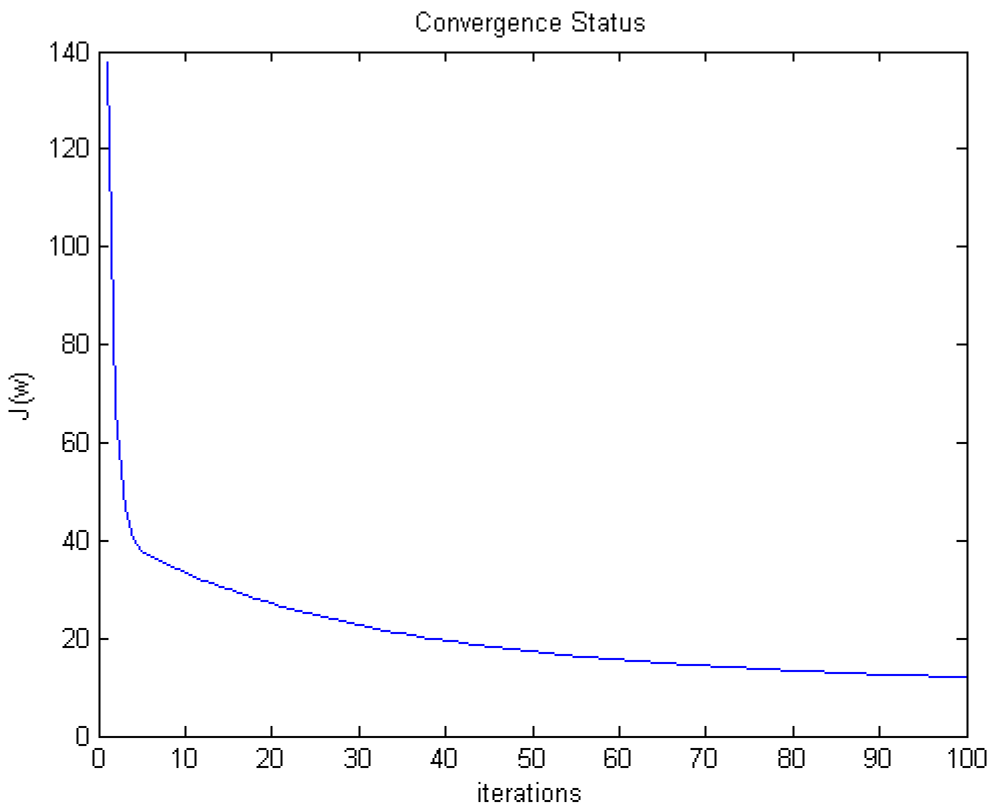
**SUHAS S**
**(SC14M081)**

# Contents

# 1.Linear Regression

## 1.1 Program 1: Program Effort Data

K value = 4 (no of folds)
Maximum iterations = 100
(c) J(w) for each iteration
137.6333 66.7635 46.5829 40.3563 37.9888 36.7056 35.7433 34.8913 34.0908 33.3249 32.5881 31.8781 31.1937 30.5338 29.8974 29.2835 28.6914 28.1201 27.5690 27.0371 26.5239 26.0285 25.5503 25.0886 24.6428 24.2124 23.7966 23.3950 23.0070 22.6321 22.2699 21.9197 21.5812 21.2540 20.9376 20.6315 20.3355 20.0491 19.7720 19.5038 19.2442 18.9928 18.7495 18.5138 18.2855 18.0643 17.8499 17.6421 17.4407 17.2454 17.0560 16.8723 16.6940 16.5211 16.3532 16.1901 16.0318 15.8781 15.7287 15.5836 15.4425 15.3054 15.1721 15.0424 14.9162 14.7934 14.6739 14.5576 14.4444 14.3340 14.2266 14.1218 14.0198 13.9202 13.8232 13.7285 13.6362 13.5461 13.4581 13.3722 13.2884 13.2065 13.1264 13.0482 12.9717 12.8970 12.8239 12.7523 12.6824 12.6139 12.5469 12.4812 12.4169 12.3540 12.2923 12.2318 12.1726 12.1145 12.0575 12.0016

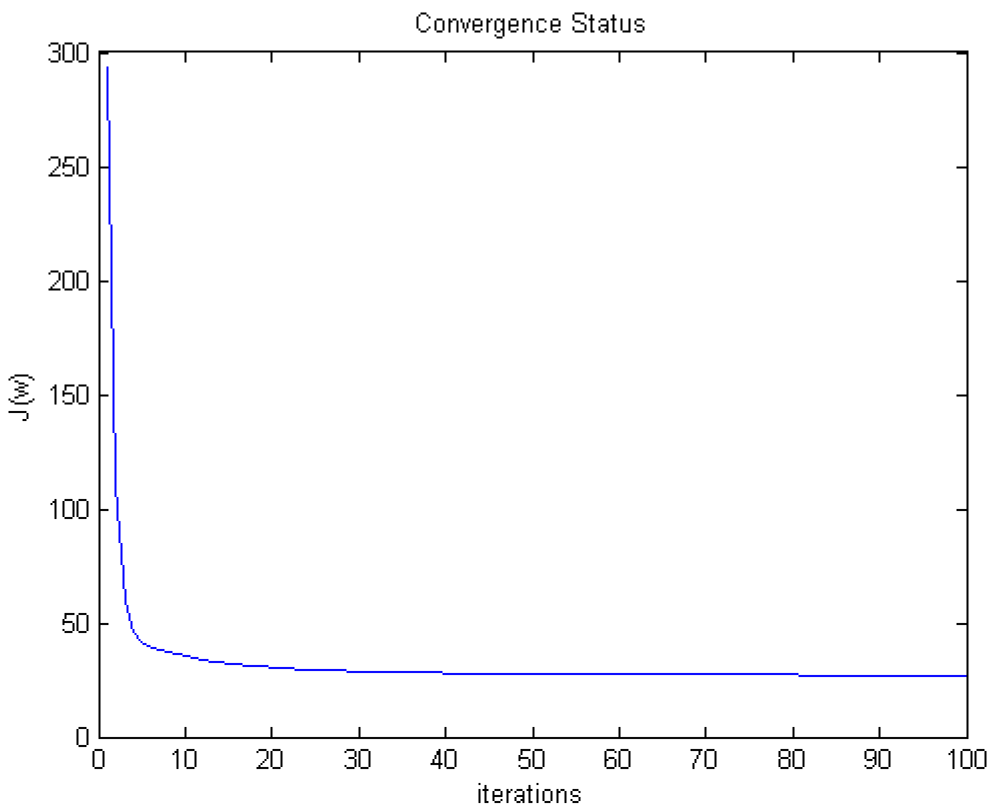



(d)model parameters
weight values : $\theta_0$ = -12.9920 $\theta_1$ = 29.2833 $\theta_2$ = 36.9038
learning rate $\alpha$ = 0.95
(e)performance of the model
mean error = 21.38

## 1.2 Program 2: Boston Housing data Data

K value = 4 (no of folds)
Maximum iterations = 100

(c) J(w) for each iteration
293.6397 109.8464 60.2526 46.1273 41.4506 39.3577 38.0306 36.9820 36.0756 35.2688 34.5440 33.8910 33.3020 32.7706 32.2908 31.8576 31.4661 31.1123 30.7924 30.5029 30.2408 30.0034 29.7882 29.5930 29.4158 29.2548 29.1083 28.9749 28.8534 28.7425 28.6411 28.5484 28.4635 28.3855 28.3139 28.2480 28.1872 28.1310 28.0789 28.0307 27.9858 27.9440 27.9050 27.8684 27.8341 27.8019 27.7715 27.7428 27.7155 27.6897 27.6651 27.6416 27.6192 27.5977 27.5770 27.5571 27.5379 27.5193 27.5014 27.4839 27.4670 27.4505 27.4344 27.4187 27.4033 27.3883 27.3736 27.3591 27.3449 27.3310 27.3172 27.3037 27.2904 27.2772 27.2643 27.2515 27.2388 27.2263 27.2139 27.2016 27.1895 27.1775 27.1656 27.1538 27.1422 27.1306 27.1191 27.1077 27.0964 27.0852 27.0741 27.0630 27.0520 27.0411 27.0303 27.0196 27.0089 26.9983 26.9878 26.9773




(d)model parameters
weight values : $\theta_0$ = 24.5337 $\theta_1$ = -1.4934 $\theta_2$ = 8.6389 $\theta_3$ = -1.8059 $\theta_4$ = 0.0249 $\theta_5$ = -0.0088 $\theta_6$ = -0.0088 $\theta_7$ = 0.7257 $\theta_8$ = -4.9007 $\theta_9$ = 0.3858 $\theta_{10}$ = -0.4787 $\theta_{11}$ = -14.6055 $\theta_{12}$ = -0.3076 $\theta_{13}$ = 13.3320 $\theta_{14}$ = -3.6732

learning rate $\alpha$ = 0.95
(e)performance of the model
mean error = 32.79

The preprocessing that i did on the data are
1.**adding bias to the feature matrix X**.
it is done as follows

```
[m n] = size(X);
X = [ones(m,1) X];
```

2.**feature scaling**

The basic idea of feature scaling is to make sure that features are on a similiar scale. This will in turn results in faster convergence of gradient descent[1]. The common technique is to make every feature in the range $0 \leq X_i \leq 1$. Feature scaling for a vector x is done as follows

$$x_i = \frac{x_i - min(x)}{max(x) - min(x)}$$

The code that performs the feature scaling is shown below.

```
function X = featureScale(x)

    [m,n]=size(x);
    maxval=max(max(x));
    minval=min(min(x));
    difference=maxval-minval;
    for i=1:m,
        for j=2:n,
            x(i,j)=(x(i,j)-minval)/difference;
        end;
    end;
    X=x;

end
```

1.Feature Scaling

The other functions that i have defined for implementing the linear regression is listed below.

1.The main function

```
function  gradient()

    maxiter=100;
    alpha=.95;  % choosen after randomly trying many values
    epsilon=1e-5;
    jval=0.0;
    prompt='K value : ';
    k=input(prompt);
    X=load('housing.txt');

    [m,n]=size(X);
    bias=ones(m,1);
    X=[bias X];   % adding bias to the X matrix
    [m,n]=size(X);
    Y = X(:,n);
    X(:,n)=[];
    X=featureScale(X);
    X=[X Y];
    [m,n]=size(X);
    Theta=zeros(k,n-1);
    jval=zeros(1,k);
    grad=zeros(k,n-1);
    testerror=zeros(1,k);
    cost=zeros(maxiter,k);

    set=cvpartition(m,'kfold',k);

    for i=1:maxiter,
        for j=1:k,
```

```
            ip=X(training(set,j),:);
            op=ip(:,n);
            ip(:,n)=[];
            [jval(j),grad(j,:)]=costFunction(ip,Theta(j,:),op);
            cost(i,j)=jval(j);
            if jval(j)<epsilon,
                break;
            end;
            Theta(j,:)=Theta(j,:)-alpha*grad(j,:);   % vectorised implementation
            testip=X(test(set,j),:);
            testop=testip(:,n);
            testip(:,n)=[];
            testerror(j)=testSetError(testip,Theta(j,:),testop);
        end;
    end;

    [minjval,idx]=min(jval);
    yval=cost(:,idx)';
    fprintf('J(w)values \n');
    disp(yval);
    avg_error=mean(cost(:,idx));
    xval=1:maxiter;
    h=figure;
    plot(xval,yval);
    xlabel('iterations');
    ylabel('J(w)');
    title('Convergence Status');
    fprintf('model parameters\n Weight values \n');
    disp(Theta(idx,:)');
    fprintf('alpha value :%0.2f \n',alpha);
    fprintf('model performance :%0.2f\n',avg_error);

end
```

2.main function

2.The cost function

```
function [jval,grad] = costFunction(X,T,y)

    [m,n]=size(X);
    jval=0.0;
    hyp=zeros(1,m);
    grad=zeros(1,n);

    for i=1:m,
        hyp(i)=X(i,:)*T';  % h(x^(i)) = T'*X[i] getting the hypothesis
        jval=jval+(hyp(i)-y(i))^2;  % h(x^(i)-y(i))^2
    end;
    jval=jval/(2*m);

    for j=1:n,
        for i=1:m,
            grad(j)=grad(j)+(hyp(i)-y(i))*X(i,j);  % h(x^(i)-y(i))*x[i][j] = grad(j)
        end;
        grad(j)=grad(j)/m;
    end;

end
```

3.cost function

3.Function to determine error in the test set

```
function err = testSetError(X,T,y)

    [m,n]=size(X);
    err=0.0;
    hyp=zeros(1,m);
    for i=1:m,
        hyp(i)=X(i,:)*T';  % h(x^(i)) = T'*X[i] getting the hypothesis
        err=err+(hyp(i)-y(i))^2;  % h(x^(i)-y(i))^2
    end;
    err=err/(2*m);

end
```

4.test set error

# 2.Maximum Likelihood Estimation

In statistics, maximum-likelihood estimation(MLE) is a method of estimating the parameters of a statistical model. When applied to a data set and given a statistical model, maximum-likelihood estimation provides estimates for the model‘s parameters[4].
The method of maximum likelihood corresponds to many well-known estimation methods in statistics. For example, one may be interested in the heights of adult female penguins, but be unable to measure the height of every single penguin in a population due to cost or time constraints. Assuming that the heights are normally(Gaussian) distributed with some unknown mean and variance, the mean and variance can be estimated with MLE while only knowing the heights of some sample of the overall population. MLE would accomplish this by **taking the mean and variance as parameters and finding particular parametric values that make the observed results the most probable**.

## 2.1 Problem Statement

Suppose we have a random sample $X_1, X_2, \ldots, X_n$ whose assumed probability distribution depends on some unknown parameter $\theta$. Our primary goal here will be to find a point estimator $\mu(X_1, X_2, \ldots, X_n)$, such that $\mu(x_1, x_2, \ldots, x_n)$ is a "good" point estimate of $\theta$, where $x_1, x_2, \ldots, x_n$ are the observed values of the random sample. For example, if we plan to take a random sample $X_1, X_2, \ldots, X_n$ for which the $X_i$ are assumed to be normally distributed with mean $\mu$ and variance $\sigma^2$, then our goal will be to find a good estimate of $\mu$ , say, using the data $x_1, x_2, \ldots, x_n$ that we obtained from our specific random sample.

## 2.2 The Basic Idea

It seems reasonable that a good estimate of the unknown parameter $\theta$ would be the value of $\theta$ that **maximizes the probability,that is, the likelihood of getting the data we observed**.So, that is, in a nutshell, the idea behind the method of maximum likelihood estimation. But how would we implement the method in practice? Well, suppose we have a random sample $X_1, X_2, \ldots, X_n$ for which the probability density (or mass) function of each $X_i$ is $f(x_i;\theta)$. Then, the joint probability mass (or density) function of $X_1, X_2, \ldots, X_n$, which we‘ll call L($\theta$) is:

$$L(\theta) = P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = f(x_1;\theta) \cdot f(x_2;\theta) \cdots f(x_n;\theta) = \prod_{i=1}^{n} f(x_i;\theta)$$

The first equality is of course just the definition of the joint probability mass function. The second equality comes from that fact that we have a random sample, which implies by definition that the $X_i$ are independent. And, the last equality just uses the shorthand mathematical notation of a product of indexed terms. Now, in light of the basic idea of maximum likelihood estimation, one reasonable way to proceed is to treat the "likelihood function" L($\theta$) as a function of $\theta$, and find the value of $\theta$ that maximizes it[2].

## 2.3 Example

Suppose we have a random sample $X_1, X_2, \ldots, X_n$ where:

- $X_i = 0$ if a randomly selected student does not own a bike, and
- $X_i = 1$ if a randomly selected student does own a bike.

Assuming that the $X_i$ are independent Bernoulli random variables with unknown parameter $p$, find the maximum likelihood estimator of $p$, the proportion of students who own a bike.
**Solution**. If the $X_i$ are independent Bernoulli random variables with unknown parameter $p$, then the probability mass function of each $X_i$ is:

$$f(x_i; p) = p^{x_i}(1-p)^{1-x_i}$$

for $x_i = 0$ or 1 and 0< $p$ <1. Therefore, the likelihood function L($p$) is, by definition:

$$L(p) = \prod_{i=1}^{n} f(x_i; p) = p^{x_1}(1-p)^{1-x_1} \times p^{x_2}(1-p)^{1-x_2} \times \cdots \times p^{x_n}(1-p)^{1-x_n}$$

for 0< $p$ <1. Simplifying, by summing up the exponents, we get:

$$L(p) = p^{\sum x_i}(1-p)^{n-\sum x_i}$$

Now, in order to implement the method of maximum likelihood, we need to find the $p$ that maximizes the likelihood $L(p)$.in order to maximize the function, we are going to need to differentiate the likelihood function with respect to $p$. In doing so, we'll use a "trick" that often makes the differentiation a bit easier[2]. Note that the natural logarithm is an increasing function of x: That is, if $x_1 < x_2$, then $f(x_1) < f(x_2)$. That means that the value of $p$ that maximizes the natural logarithm of the likelihood function *ln(L(p))* is also the value of $p$ that maximizes the likelihood function L($p$). So, the "trick" is to take the derivative of *ln(L(p))* (with respect to $p$) rather than taking the derivative of L($p$).

In this case, the natural logarithm of the likelihood function is:

$$logL(p) = (\sum x_i)log(p) + (n - \sum x_i)log(1-p)$$

Now, taking the derivative of the log likelihood, and setting to 0, we get:

$$\frac{\partial logL(p)}{\partial p} = \frac{\sum x_i}{p} - \frac{(n - \sum x_i)}{1-p} \equiv 0$$

Now, multiplying through by p(1-p), we get:

$$(\sum x_i)(1-p) - (n - \sum x_i)p = 0$$

$$\Rightarrow \sum x_i - np = 0$$

solving for $p$,

$$\hat{p} = \frac{\sum\limits_{i=1}^{n} x_i}{n}$$

where $\hat{p}$ is used to indicate that it is an estimate.

or, alternatively, an estimator:

$$\hat{p} = \frac{\sum_{i=1}^{n} X_i}{n}$$

### 2.3 Definition

Now, with that example behind us, let us take a look at formal definitions of the terms **(1) likelihood function, (2) maximum likelihood estimators, and (3) maximum likelihood estimates**.

Let $X_1, X_2, \dots, X_n$ be a random sample from a distribution that depends on one or more unknown parameters $\theta_1, \theta_2, \dots, \theta_m$ with probability density (or mass) function $f(x_i; \theta_1, \theta_2, \dots, \theta_m)$. Suppose that $(\theta_1, \theta_2, \dots, \theta_m)$ is restricted to a given parameter space $\Omega$. Then:

(1) When regarded as a function of $\theta_1, \theta_2, \dots, \theta_m$, the joint probability density (or mass) function of $X_1, X_2, \dots, X_n$:

$$L(\theta_1, \theta_2, \dots, \theta_m) = \prod_{i=1}^{n} f(x_i; \theta_1, \theta_2, \dots, \theta_m)$$

($(\theta_1, \theta_2, \dots, \theta_m)$ in $\Omega$) is called the **likelihood function**.

(2)if:

$$[u_1(x_1, x_2, \dots, x_n), u_2(x_1, x_2, \dots, x_n), \dots, u_m(x_1, x_2, \dots, x_n)]$$

is the m-tuple that maximizes the likelihood function, then:

$$\hat{\theta}_i = u_i(X_1, X_2, \dots, X_n)$$

is the **maximum likelihood estimator** of $\theta_i$, for $i = 1, 2, \dots, m$.

(3) The corresponding observed values of the statistics in (2), namely:

$$[u_1(x_1, x_2, \dots, x_n), u_2(x_1, x_2, \dots, x_n), \dots, u_m(x_1, x_2, \dots, x_n)]$$

are called the **maximum likelihood estimates** of $\theta_i$, for $i = 1, 2, \dots, m$.

## References

[1] Machine Learning - Coursera, *https://class.coursera.org/ml-005/lecture?lecture_player=html5*

[2] Maximum Likelihood Estimation *https://onlinecourses.science.psu.edu/stat414/node/191*.

[3] *http://mathworld.wolfram.com/LikelihoodFunction.html*

[4] Maximum Likelihood Estimation, wikipedia - *1.http://en.wikipedia.org/wiki/Maximum_likelihood*